

|  |
| --- |
| Introduction to Maven  Practice/home work |
|  |

| Related Artifacts | |
| --- | --- |
| Ref. | Name |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Abbreviations and Acronyms | |
|  |  |
|  |  |

# Creating a simple project

* Create a simple Maven project using maven-archetype-quickstart archetype
* Examine project structure and pom.xml file
* Give answers to the following questions:
  + What is the scope of junit dependency?

**Answer**: Test is the scope of junit dependency

* + Is it possible to change junit dependency scope? How you could change the scope?

**Answer**: Yes, it’s possible. Just need change value in <scope></scope>.

* + Is it possible to have source code into, for example, src/source folder?

**Answer**: Yes, it’s possible. Need to add depencies.

1. Please refer to <http://maven.apache.org/archetypes/maven-archetype-quickstart/> for archetype usage details.

# Explore Your Project

* Generate effective pom.xml file using mvn help:effective-pom
* Examine effective pom.xml
* Give answers to the following questions:
  + Why the effective pom.xml is different from your pom.xml?

**Answer**: It has many elements that not included in pom.xml

* + Is it possible to modify effective pom.xml?

**Answer**: No, because it autogenerated

* + What is maven-jar-plugin plugin used for? When it will be executed?

**Answer**: It describes default lifecycle bindings (in this case – jar). It will be executed at package phase.

<plugin>

<artifactId>maven-jar-plugin</artifactId>

<version>2.4</version>

<executions>

<execution>

<id>default-jar</id>

<phase>package</phase>

<goals>

<goal>jar</goal>

</goals>

</execution>

</executions>

</plugin>

# Adding new dependencies

* Modify your main class App.java with following code.

1. Please keep package declaration as is.

import org.apache.logging.log4j.Logger;

import org.apache.logging.log4j.LogManager;

public class App

{

final static Logger logger = LogManager.getLogger(App.class);

public static void main( String[] args )

{

logger.info("This is info");

logger.warn("This is warn");

logger.error("This is error");

logger.fatal("This is fatal");

}

}

* Add Log4j dependency into your pom.xml
* Please make sure to use Log4j with the latest version

groupId - org.apache.logging.log4j

artifactId - log4j-core

1. Refer to <https://mvnrepository.com> to find the latest version

* Build your project
* Examine target folder
* Give answers to the following questions:
  + Where is build artifacts are located?

**Answer**:They located in target folder.

* + What is the scope of log4j-core dependency?

**Answer**: Compile, provided, test.

* + Where is log4j jar located? What is .m2 folder and where it is located?

**Answer**:C:\Users\Vladimir\_Polovnev\.m2\repository\org\apache\logging\log4j\log4j-core\2.11.2\log4j-core-2.11.2.jar .m2 is a local maven repository

# Build dependency tree

* Run mvn dependency:resolve
* Run mvn dependency:tree
* Examine previous commands output
* Give answers to the following questions:
  + How dependencies are resolved?

**Answer**: Dependencies are resolved in test scope.

* + Why we got log4j-api as a dependency into tree?

**Answer**: Because mvn dependency:tree show hierarchical list of dependencies, so it show transit dependencies.

# Using maven-assembly-plugin

* Examine maven-assembly-plugin documentation <https://maven.apache.org/plugins/maven-assembly-plugin/usage.html>
* Bind a plugin to a package phase
* Add the following configuration for assembly plugin

<configuration>

<descriptorRefs>

<descriptorRef>jar-with-dependencies</descriptorRef>

</descriptorRefs>

<archive>

<manifest>

<mainClass>com.demo.App</mainClass>

</manifest>

</archive>

</configuration>

1. com.demo.App which is main class location might be different into your setup. Please verify package structure

* Build your project with mvn package
* Examine target folder and find a file with -jar-with-dependencies.jar suffix
* Run it with java –jar <jar name here>

1. Don’t forget to replace <jar name here> with actual file name
2. If no file was generated, please make sure that you bind a plugin to the package phase

# Working with resources

* Create a file named log4j2.xml with the following content

<?xml version="1.0" encoding="UTF-8"?>

<Configuration status="warn">

<Properties>

<Property name="basePath">logs</Property>

</Properties>

<Appenders>

<RollingFile name="fileLogger" fileName="${basePath}/app-info.log" filePattern="${basePath}/app-info-%d{yyyy-MM-dd}.log">

<PatternLayout>

<pattern>[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n</pattern>

</PatternLayout>

<Policies>

<TimeBasedTriggeringPolicy interval="1" modulate="true" />

</Policies>

</RollingFile>

<Console name="console" target="SYSTEM\_OUT">

<PatternLayout pattern="[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n" />

</Console>

</Appenders>

<Loggers>

<Logger name="main.logger" level="debug" additivity="true">

<appender-ref ref="fileLogger" level="debug" />

</Logger>

<Root level="debug" additivity="false">

<appender-ref ref="console" />

</Root>

</Loggers>

</Configuration>

* Put this file into src/main/resources folder (create this folder if needed)
* Run mvn package to create a package
* Examine target folder and find a file with -jar-with-dependencies.jar suffix
* Run it with java –jar <jar name here>
* Examine differences between program output into step 5 and step 6
* Give answers to the following questions:
  + What is resources folder?

**Answer**: from that directory files will be copied to the build output.

* + Where files from resources folder will be located after build?

**Answer**: target/classes.

* + Is there any possibility to modify default resource management behavior?

**Answer**: Yes, for example we can specify resource directories. Also, we can adding multiple <resource> elements.

1. Please refer to Apache Maven Resources Plugin documentation page <https://maven.apache.org/plugins/maven-resources-plugin/> in order to get some additional information.

Quiz

1. You could verify your quiz results online <https://www.tutorialspoint.com/maven/maven_online_quiz.htm> but please be honest and answer to this questions before getting to an answer from the site provided.
2. What of the following is true about POM?

* It is fundamental Unit of Work in Maven.
* It is an XML file.
* Both of the above. **+**
* None of the above.

1. Which of the following configuration element is present in POM.xml?

* developers
* mailing list
* Both of the above. **+**
* None of the above.

1. Which of the following phase in maven life cycle compiles the source code of the project?

* validate
* compile **+**
* test
* package

1. Which of the following is true about Maven repository?

* A repository is a place i.e. directory where all the project jars, library jar, plugins or any other project specific artifacts are stored.
* Maven repository is used by Maven.
* Both of the above. **+**
* None of the above.

1. Which of the following maven plugin executes during the build and should be configured in the element of pom.xml?

* Build plugins **+**
* Reporting plugins
* Both of the above
* None of the above.

1. Which of the following is the default value of packaging?

* ear
* war
* jar **+**
* pom

1. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

* true **+**
* false

1. Which of the following is correct about super POM?

* Maven use the effective pom (configuration from super pom plus project configuration) to execute relevant goal.
* It helps developer to specify minimum configuration detail in his/her pom.xml. Although configurations can be overridden easily.
* Both of the above. **+**
* None of the above.

1. Which of the following phase in maven life cycle compiles the test source code into the test destination directory?

* test-compile **+**
* process-test-sources
* generate-test-sources
* None of the above.

1. Which of the following phase in maven life cycle processes and deploys the package if necessary into an environment where integration tests can be run?

* process-resources
* pre-integration-test
* integration-test **+**
* None of the above.